APPENDIX

```
input = 0;
output = 1;
for (i = 0; i < num_inputs; i++){
     C[input][i] = input_capacity[i];
     for (j = 0; j < num_outputs; j++)
          C[input][i] -= demanded_rates[i][j];}
for (j = 0; j < num_outputs; j++){
     C[output][j] = output_capacity[j];
     for (i = 0; i < num_inputs; i++)
          C[output][j] -= demanded_rates[i][j];}
for (j = 0; j < num_outputs; j++)
     x[output][j] = 0.0;
for (i = 0; i < num_inputs; i++)
     for (j = 0; j < num_outputs; j++)
          D[i][j] = desired_rates[i][j];
for (k = 1; k <= num_global_iterations; k++){
     for (i = 0; i < num_inputs; i++){
          for (j = 0; j < num_outputs; j++){
               w[j] = weights[i][j];
               d[j] = D[i][j];}
     x[input][i] = dist(num_inputs, w, d, C[input][i],
r);
          if (k == num_global_iterations)
          for (j = 0; j < num_outputs; j++)
               requested_rates[i][j] = r[j];
```

```
        else{
             freeze = 1;
             if (x[input][i] != inf)
                  for (j = 0; j < num_outputs; j++)
                       if (x[output][j] != inf
                        && x[input][i] > x[output][j])
                            freeze = 0;
                  if (freeze){
                       x[input][i] = inf;
                       for (j = 0; j < num_outputs; j++)
                            D[i][j] = r[j];}}}
        if (k < num_global_iterations){
        for (j = 0; j < num_outputs; j++){
                  for (i = 0; i < num_inputs; i++){
                       w[i] = weights[i][j];
                       d[i] = D[i][j];}
             x[output][j]   =   dist(num_outputs,   w,   d,
        C[output][j], r);
                  freeze = 1;
             if (x[output][j] != inf)
                  for (i = 0; i < num_inputs; i++)
                       if (x[input][i] != inf
                        && x[output][j] > x[input][i])
                            freeze = 0;
                  if (freeze){
                       x[output][j] = inf;
                       for (i = 0; i < num_inputs; i++)
                            D[i][j] = r[i];}}}}
   for (j = 0; j < num_outputs; j++){
        for (i = 0; i < num_inputs; i++){
```

```
            w[i] = weights[i][j];
                                d[i] = requested_rates[i][j];}
        dist(num_outputs, w, d, C[output][j], r);
        for (i = 0; i < num_inputs; i++)
             allocated_rates[i][j] = r[i];}
```